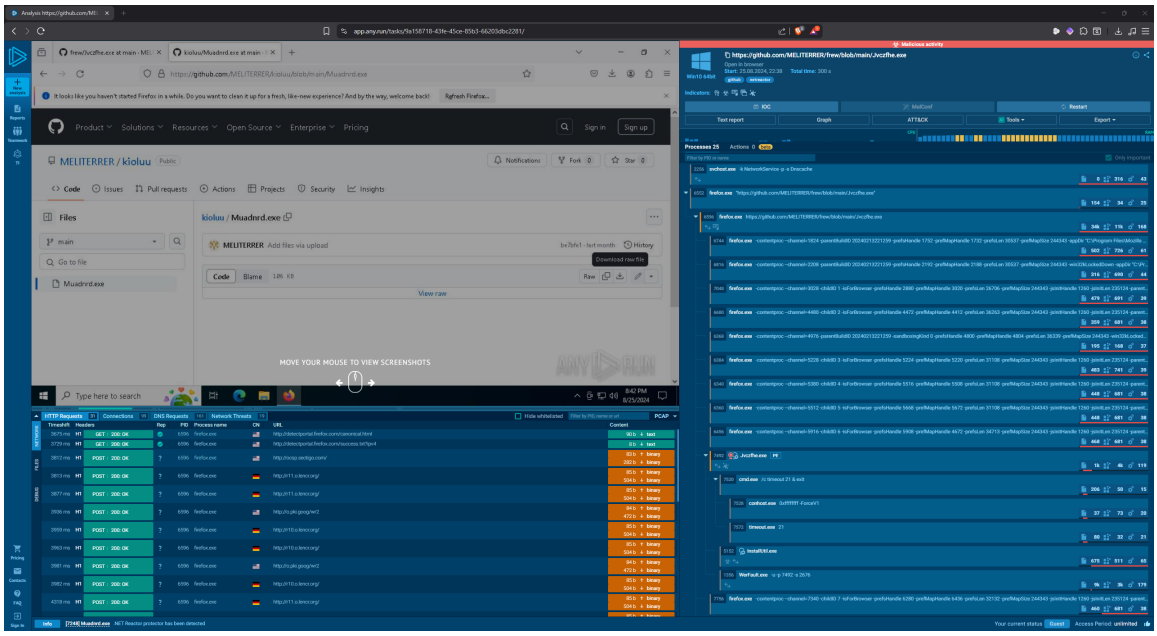
Report Tecnico:
Analisi Malware e Studio IoC

Eseguibili Jvczfhe.exe / Muadnrd.exe

Autore: Josh Van Edward D Abanico
Data: 21 febbraio 2026



Obiettivo: Analisi comportamentale ed estrazione IoC
Metodologia: Sandbox Analysis (ANY.RUN)

Indice

1 Introduzione e Scenario

1.1 Obiettivi

L'obiettivo di questa attività è analizzare un'infezione malware partendo dal suo vettore iniziale, tracciarne l'albero di esecuzione (Process Tree) e identificare gli Indicatori di Compromissione (IoC). L'analisi mira a comprendere le tecniche di evasione, persistenza e comunicazione utilizzate dal codice malevolo.

1.2 Ambiente di Test

L'analisi è stata condotta in ambiente isolato tramite la piattaforma di sandboxing **ANY.RUN**, su una macchina virtuale Windows 10 (Build 19041).

2 Fase 1: Vettore d'Infezione e Download

L'infezione ha inizio tramite l'utilizzo del browser web `firefox.exe`, dal quale viene scaricato un file eseguibile ospitato su un repository GitHub pubblico.

Come mostrato dai log di rete e dall'interfaccia di analisi, il file `Jvczfhe.exe` viene scaricato dal percorso `https://github.com/MELITERRER/frew/blob/main/Jvczfhe.exe`.

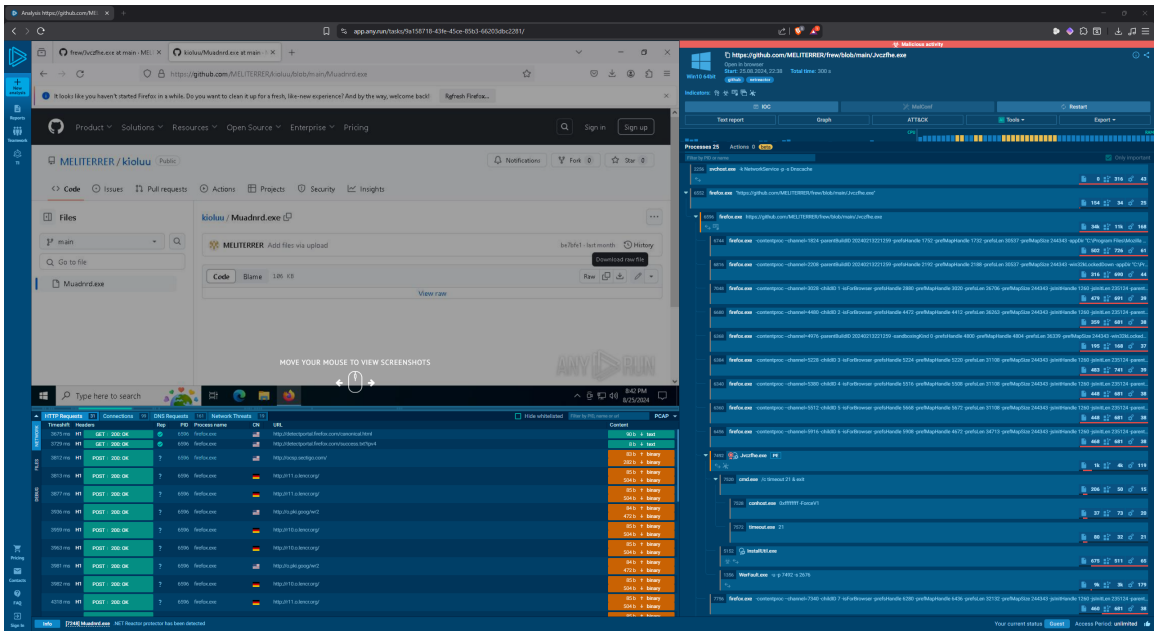


Figura 1: Interfaccia ANY.RUN con il download da GitHub

3 Analisi del Process Tree

Una volta scaricati i payload, l'albero dei processi evidenzia un flusso di esecuzione complesso, ramificato in due catene principali derivanti da `firefox.exe`:

- L'esecuzione di `jvczfhe.exe`.

- L'esecuzione di `muadnrd.exe`.

Entrambi i processi principali generano a loro volta processi figli, sfruttando utility di sistema legittime (LOLBins) per eseguire azioni malevole e ritardare l'esecuzione per evadere i controlli antivirus.

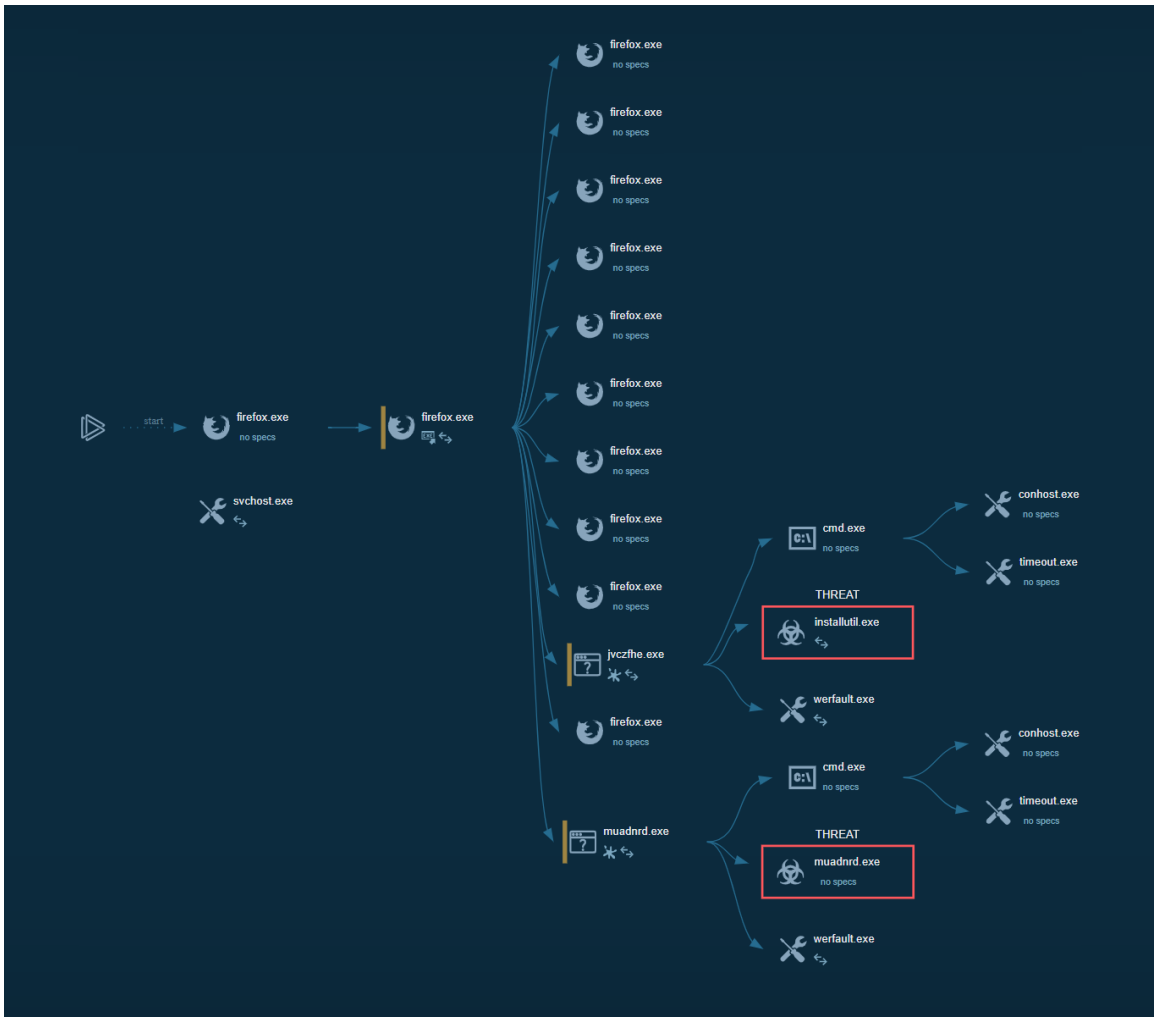


Figura 2: Process Tree che mostra le due diramazioni principali

4 Analisi Dettagliata ed Estrazione IoC

4.1 Analisi del ramo Jvczfhe.exe

L'eseguibile `Jvczfhe.exe` (PID: 7492) ha ottenuto un Threat Score di 51/100 (Suspicious). L'analisi dinamica ha rivelato una serie di comportamenti volti all'evasione e alla discovery del sistema bersaglio:

- **Evasion:** Disabilita il logging degli eventi di Windows (T1562.002).
- **Discovery:** Effettua query al registro di sistema per leggere informazioni ambientali, il GUID della macchina e le policy del software (T1012, T1082).

Il processo avvia una shell di comando (`cmd.exe`, PID: 7520) passando la stringa `"cmd" /c timeout 21 & exit`. Questo comando genera a sua volta l'esecuzione di `conhost.exe`

(PID: 7528) per la gestione della finestra della console e di `timeout.exe` (PID: 7572). L'uso di `timeout` per 21 secondi è una chiara tecnica di ritardo dell'esecuzione (Delay Execution) volta a bypassare l'analisi automatizzata delle sandbox.

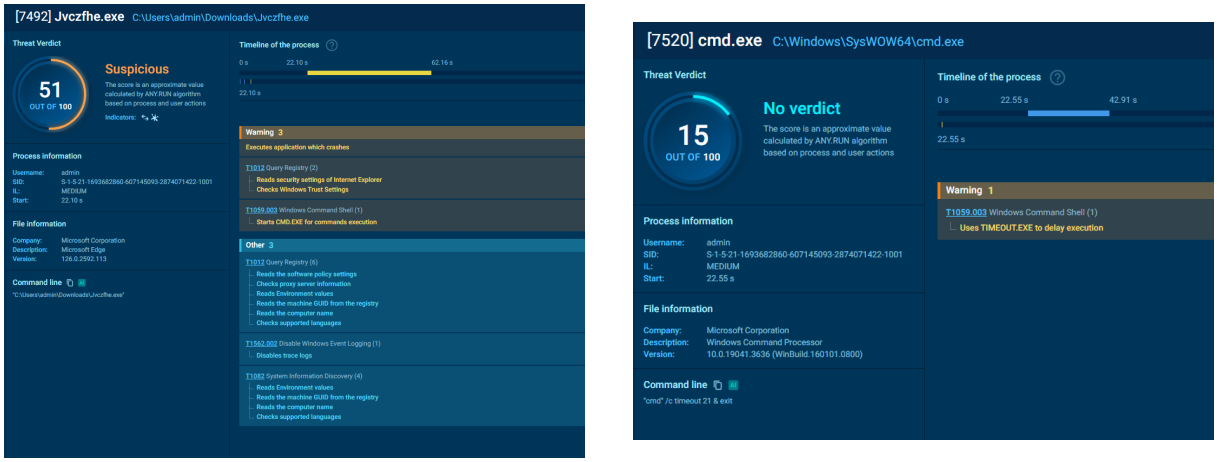


Figura 3: Dettagli del processo Jvczfhe.exe e della command line di cmd.exe

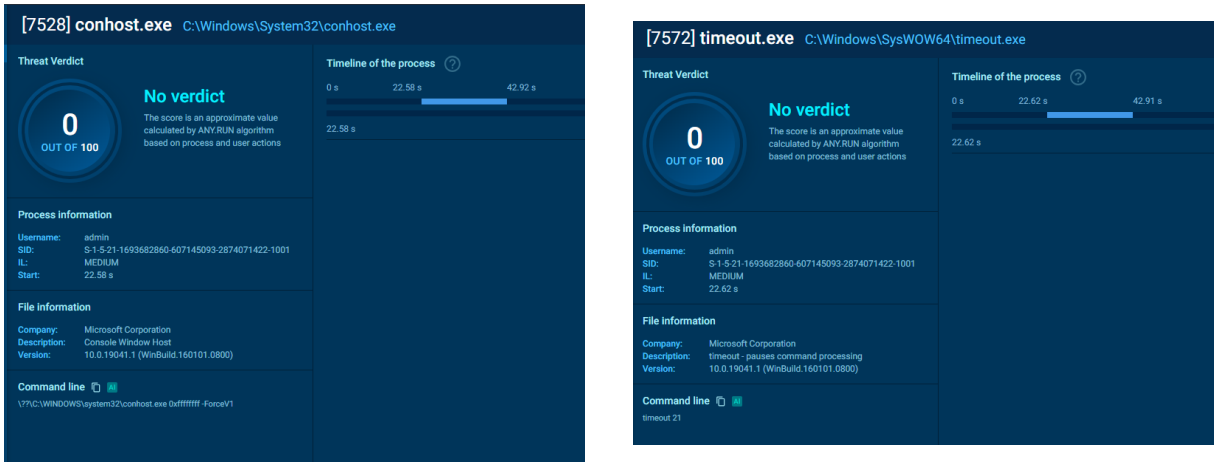


Figura 4: Esecuzione di conhost.exe e timeout.exe per generare il ritardo di 21 secondi

Successivamente, viene rilevata l'esecuzione di `InstallUtil.exe` (PID: 5152), uno strumento legittimo del .NET Framework. Il suo utilizzo in questo contesto rappresenta una classica tecnica LOLBins (Living Off The Land Binaries) per mascherare l'esecuzione di codice malevolo. L'analisi di questo processo evidenzia:

- L'uso dell'offuscatore **.NET Reactor** per proteggere il payload.
- Tentativi di connessione verso una porta non standard (T1571).
- Ulteriore raccolta di informazioni dal sistema (lettura del GUID, nome computer e lingue supportate).

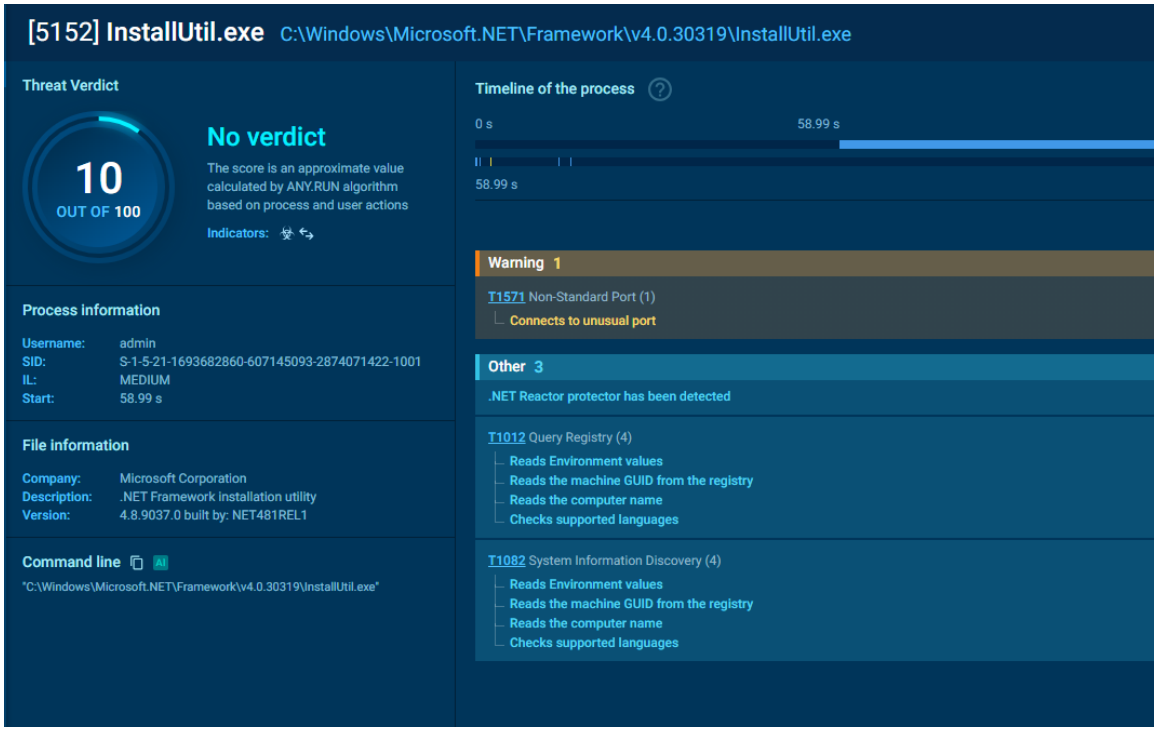


Figura 5: Analisi del processo InstallUtil.exe e rilevamento del protector .NET Reactor

L'esecuzione della catena di infezione relativa a questo ramo si conclude con il crash dell'applicativo principale (`Jvczfhe.exe`, PID 7492), evento che viene intercettato e gestito dal processo di sistema `WerFault.exe` (Windows Problem Reporting, PID: 1356).

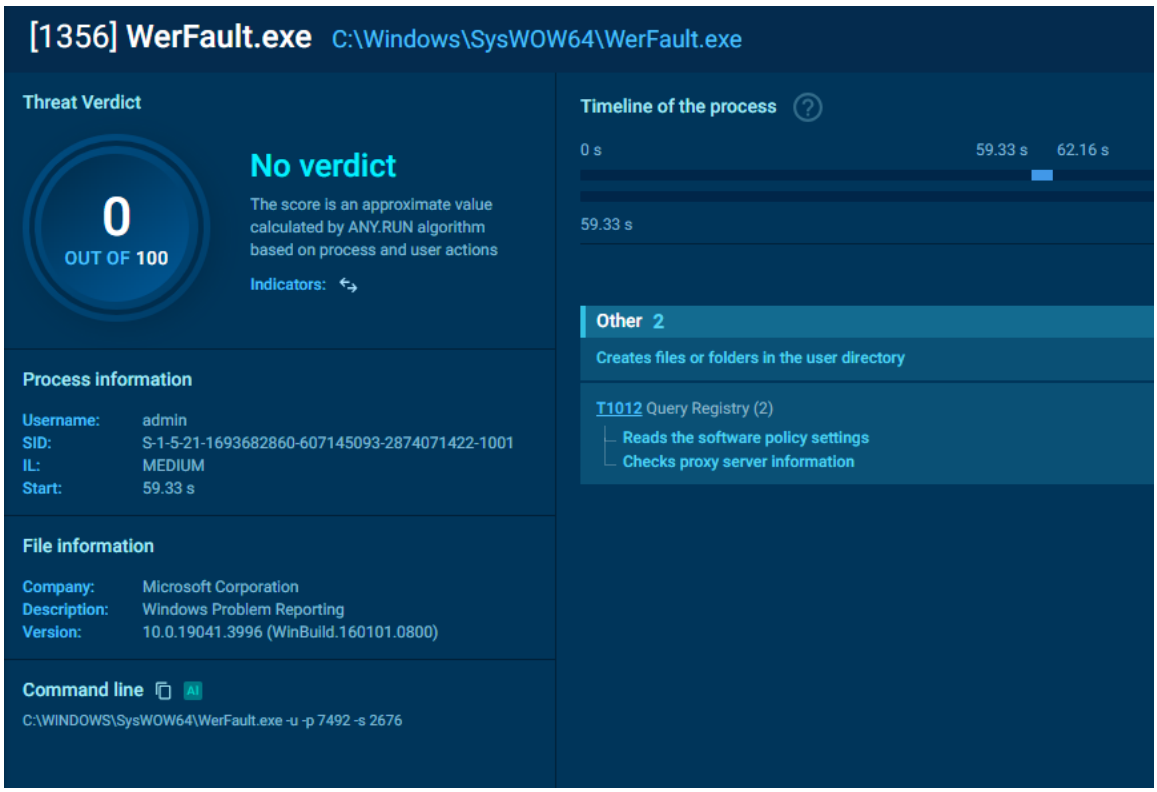


Figura 6: Intervento di WerFault.exe a seguito del crash del dropper

4.2 Analisi del ramo Muadnrd.exe

Parallelamente, il file `Muadnrd.exe` (PID: 7824) esegue operazioni speculari, ottenendo un Threat Score di 62/100. L'analisi dinamica rivela i seguenti comportamenti in ordine cronologico:

- **Discovery e Impair Defenses:** Interroga le impostazioni di sicurezza di Internet Explorer e disabilita preventivamente l'Event Logging (T1562.002).
- **Delay Execution:** Avvia una shell `cmd.exe` (PID: 7876) con il comando `/c timeout 21 & exit`. Questo processo genera a sua volta due figli:
  - `conhost.exe` (PID: 7860): gestisce l'istanza invisibile della console.
  - `timeout.exe` (PID: 7968): esegue un conto alla rovescia di 21 secondi, fungendo da meccanismo evasivo per aggirare l'analisi temporale tipica delle sandbox.
- **Execution / Unpacking:** Successivamente a questa fase di evasione temporale, `Muadnrd.exe` lancia una copia di se stesso generando un nuovo processo omonimo (PID: 7248). Una particolarità di questo comportamento è che indica una probabile tecnica di unpacking in memoria (o process hollowing). Anche su questo nuovo processo è stata rilevata la presenza dell'offuscatore **.NET Reactor**.
- **Crash:** Come nel primo ramo, il flusso termina con il crash del dropper originale (PID: 7824), evento catturato dal gestore di sistema `WerFault.exe` (PID: 7584).

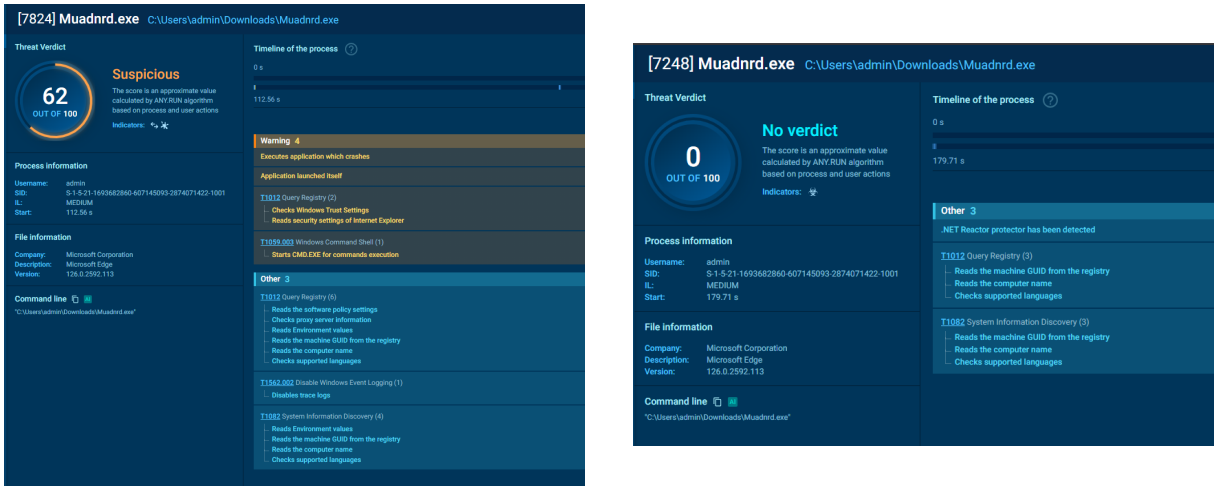


Figura 7: Dettagli dei processi Muadnrd.exe (PID 7824 e 7248) e relative attività

5 Conclusioni e Sintesi Comportamentale

L'analisi dinamica dei due campioni (`Jvczfhe.exe` e `Muadnrd.exe`) permette di classificarli inequivocabilmente come **Dropper** o **Loader** di primo/secondo stadio. Il loro obiettivo primario è stabilire un punto d'appoggio sul sistema bersaglio, aggirare i meccanismi di difesa e caricare in memoria un payload finale.

Entrambi i malware condividono una base logica e strategica quasi identica per le fasi di *Discovery* e *Defense Evasion*:

- **Fingerprinting e Discovery:** Interrogano il registro di sistema per raccogliere informazioni sull'ambiente host (GUID della macchina, nome computer, lingue supportate). Questa operazione è tipicamente utilizzata per verificare di non trovarsi all'interno di un ambiente di sandboxing o di ricerca.
- **Evasione Temporale (Time-Based Evasion):** Sfruttano la combinazione di `cmd.exe` e `timeout.exe` per inserire un ritardo artificiale di 21 secondi. L'obiettivo è esaurire il tempo di scansione assegnato dai sistemi di analisi automatizzata (come gli antivirus cloud o le sandbox) prima che il codice malevolo si riveli.
- **Inibizione delle Difese:** Disabilitano preventivamente il tracciamento degli eventi (Event Logging) di Windows per operare nell'ombra e non lasciare log (IoC) delle loro attività successive.
- **Offuscamento:** I binari risultano protetti tramite **.NET Reactor**, un packer commerciale utilizzato per rendere inefficace l'analisi statica e il reverse engineering.

La vera differenza tra i due campioni risiede nel vettore di *Execution*, ovvero nel modo in cui avviano il payload una volta superata la fase di evasione:

- **Jvczfhe.exe (Approccio LOLBins):** Delega l'esecuzione a un processo legittimo di sistema, `InstallUtil.exe`. Questo metodo, noto come *Proxy Execution*, serve a bypassare i controlli basati su whitelist (come AppLocker), ingannando il sistema operativo che vede operare un'applicazione fidata.
- **Muadnrd.exe (Approccio Process Injection):** Disimballa (unpack) il payload direttamente in memoria e genera un processo figlio con il suo stesso nome (`Muadnrd.exe` PID 7248). Questa è una chiara evidenza di tecniche come il *Process Hollowing* o il *Self-Injection*, volte a mascherare il codice malevolo facendolo girare sotto l'ombrello di un processo apparentemente identico a quello originale.

In sintesi, i due eseguibili rappresentano due diramazioni tattiche di una medesima catena di infezione, accomunate dall'hosting iniziale su GitHub e dalle tecniche evasive, ma differenziate nel metodo di rilascio del payload per massimizzare le probabilità di eludere i sistemi EDR/Antivirus dell'host vittima.